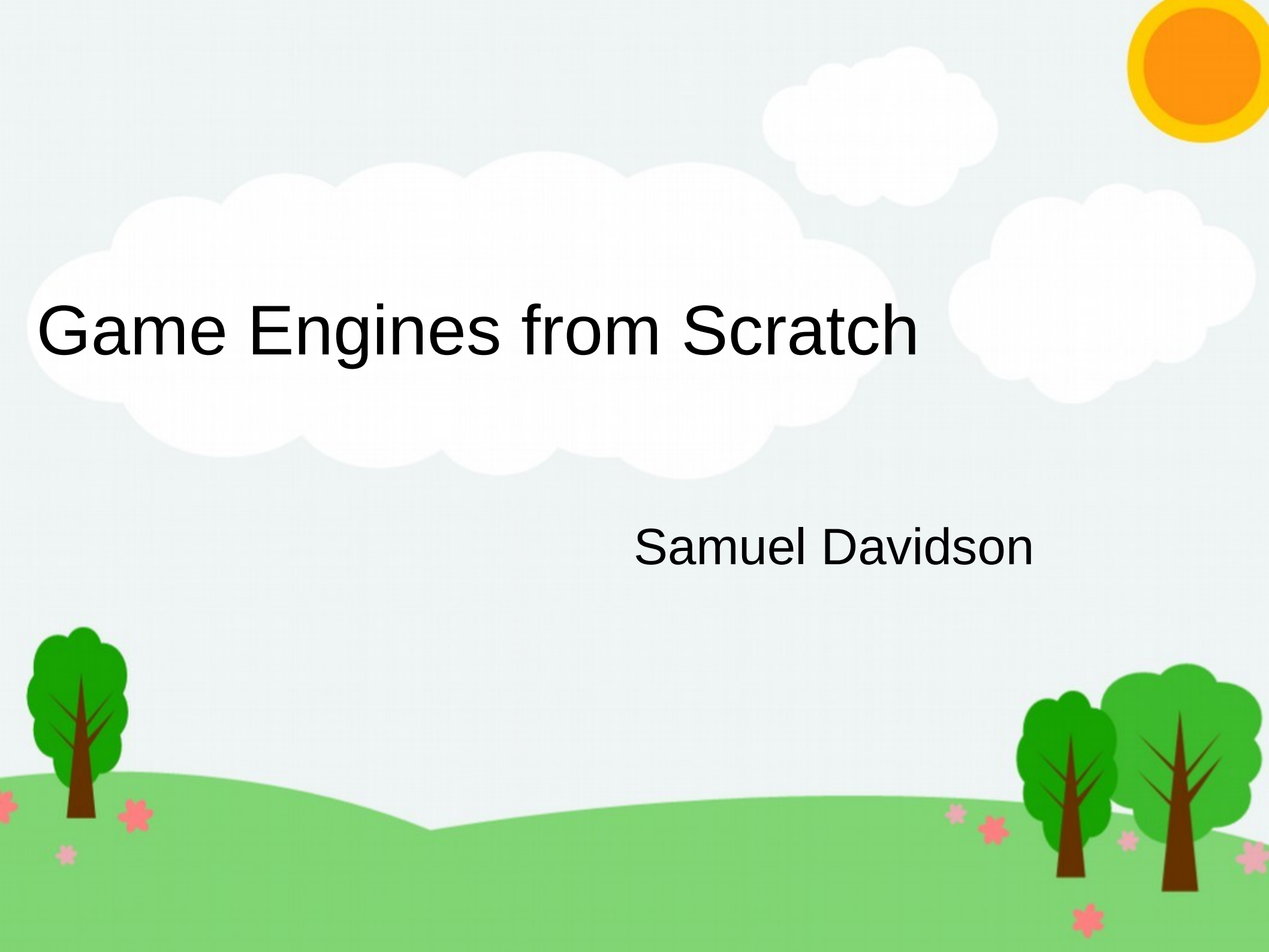
Game Engines from Scratch

Samuel Davidson

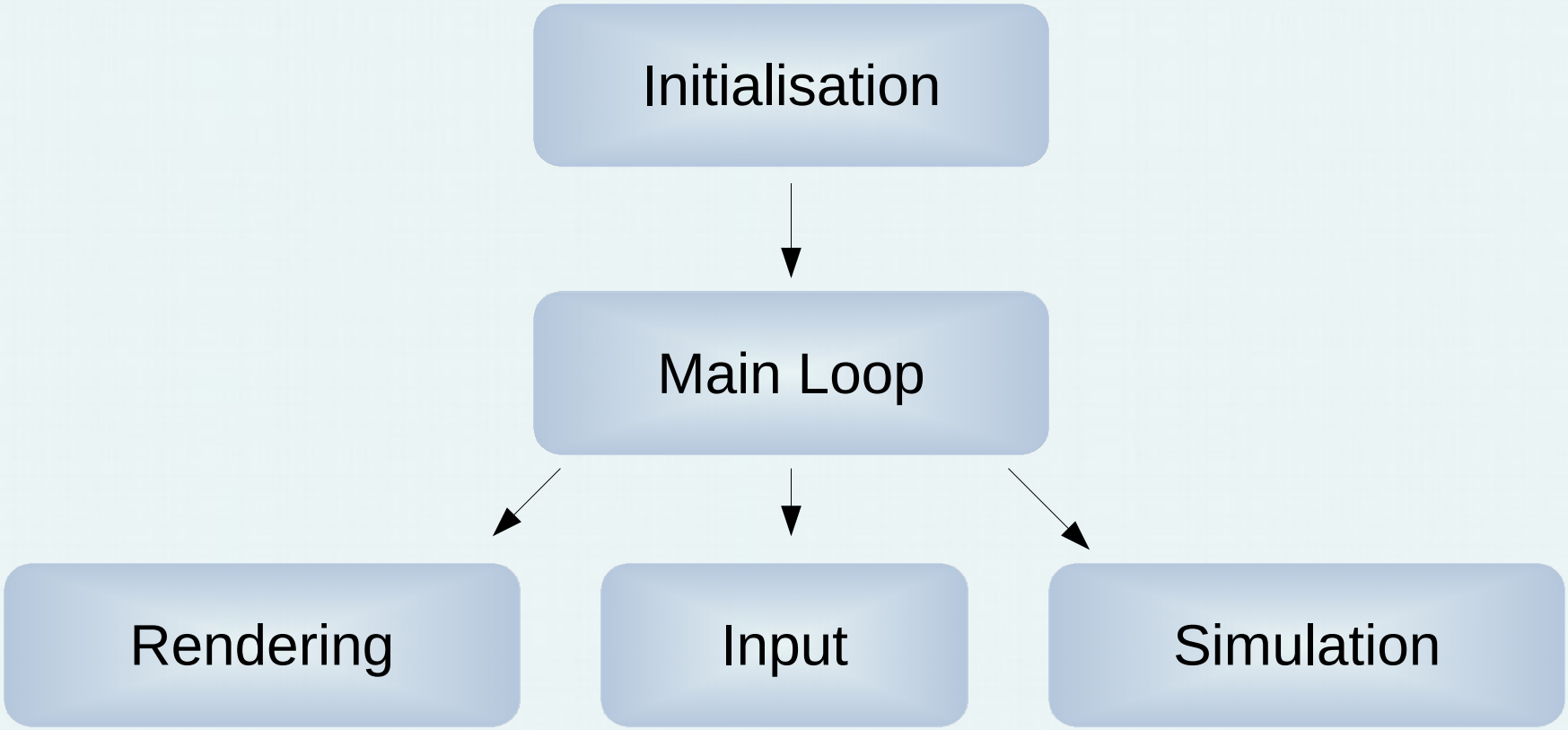


Why DIY?

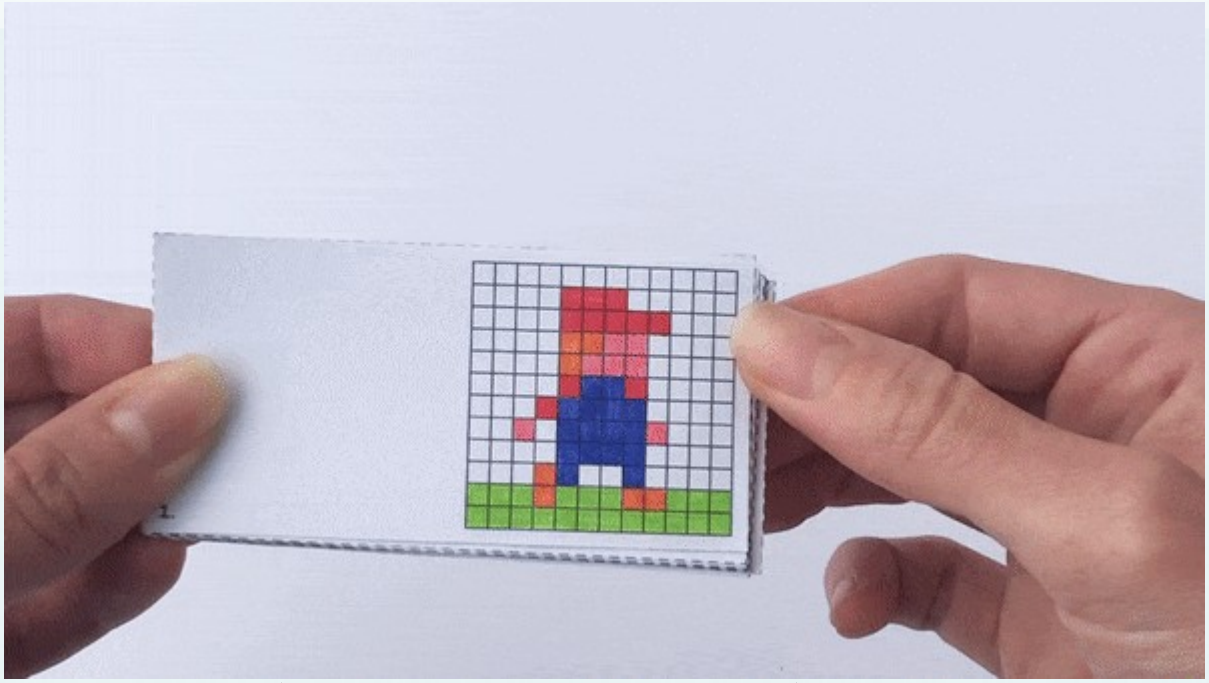
- You want to gain XP
- Portfolio piece (non/technical bridge)
- You want to build a good game
- Your game has unique requirements

  (UE5/Unity/Swarm Engine)

General Architecture



Rendering/Simulation Decouple



Rendering
- renders scene
- governs FPS (deltaTime)

Simulation
- runs whole sim
- completes each frame

Architecture Walkthrough

- Welcome to thy Jungle
- Mr Blue Square

OOP & DOD

Object Oriented Programming

```
Class Car {
    int ID;                 // 4 bytes
•   Color Colour;           // 8 bytes
•   byte Wheels;            // 1 byte
•   float BrakeForce;       // 2 bytes
•   float Velocity;         // 8 bytes
•   bool IsBraking;         // 1 bit
}


Function ApplyBrakes(Car car) {
    if (car.IsBraking) {
        car.Velocity -= car.BrakeForce;
    }
}
Array<Car> Cars = [Car1, Car2, Car3, ...];


for (int i = 0; i < Cars.length; i++) {
    ApplyBrakes(Cars[i]);
}
```

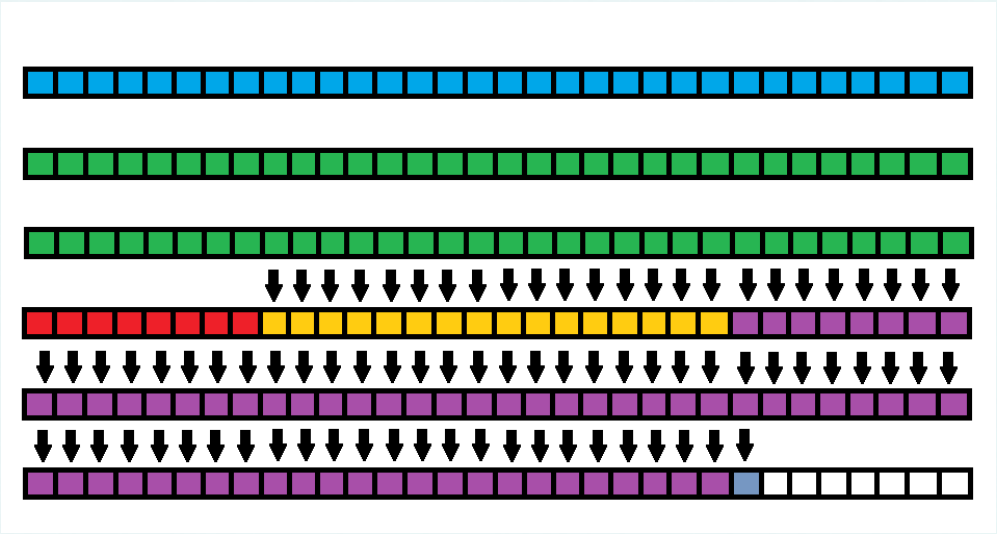
Data Oriented Design

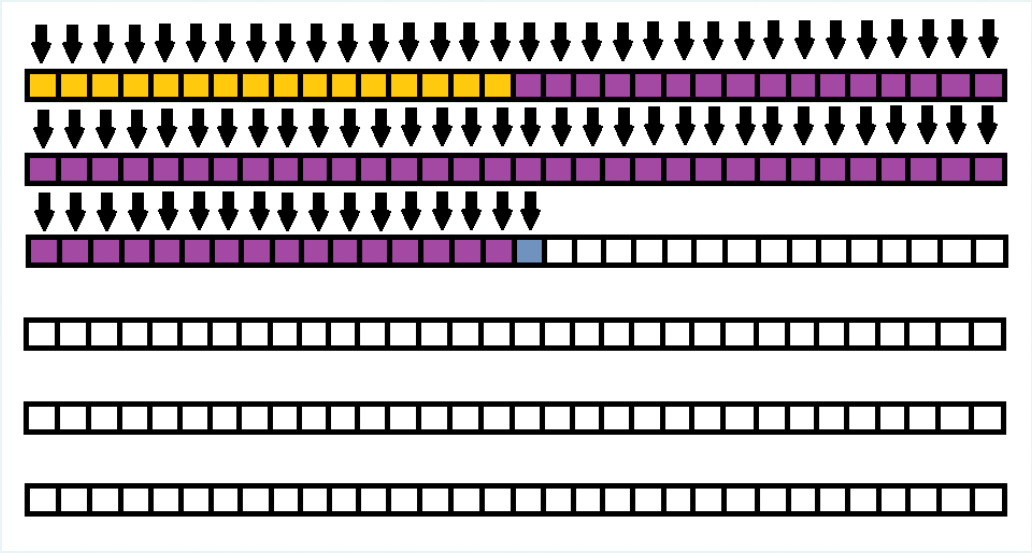
```
// ======== COMPONENT STORAGE ========
Struct Cars{
    int[] IDs;              // 4 bytes each
    Color[] Colours;        // 8 bytes each
    byte[] Wheels;          // 1 byte each
    float[] BrakeForces;    // 2 bytes each
    float[] Velocities;     // 8 bytes each
    bool[] IsBraking;       // 1 bit each
}
// ======== COMPACT WORKING SET FOR SYSTEM ========
Struct BrakingCars {
    float[] Velocities;
    float[] BrakeForces;
    int[]   OriginalIDs;    // used to write back results
}

// ======== HELPER FUNCTION TO EXTRACT COMPACT DATA ========
Function extractBrakingComponents(Cars cars) -> BrakingCars {
    BrakingCars result;
    for (int i = 0; i < cars.IDs.length; i++) {
        if (cars.IsBraking[i]) {
            result.Velocities.push(cars.Velocities[i]);
            result.BrakeForces.push(cars.BrakeForces[i]);
            result.OriginalIDs.push(i);
        }
    }
    return result;
}
// ======== BRAKE SYSTEM ========
Function ApplyBrakes(float[] velocities, float[] brakeForces) {
    for (int i = 0; i < velocities.length; i++) {
        velocities[i] -= brakeForces[i];
    }
}

// ======== WRITE RESULTS BACK ========
Function WriteVelocitiesBack(Cars cars, BrakingCars braking) {
    for (int i = 0; i < braking.OriginalIDs.length; i++) {
        int id = braking.OriginalIDs[i];
        cars.Velocities[id] = braking.Velocities[i];
    }
}
// ======== EXAMPLE USAGE ========
Cars cars = {
    IDs:        [0, 1, 2, 3],
    Colours:    [Red, Blue, Green, Yellow],
    Wheels:     [4, 4, 3, 4],
    BrakeForces: [2.0, 1.6, 2.2, 1.8],
    Velocities:  [70.0, 50.0, 60.0, 80.0],
    IsBraking:   [true, true, false, true]
};
// Pipeline:
rakingCars braking = extractBrakingComponents(cars);
ApplyBrakes(braking.Velocities, braking.BrakeForces);
WriteVelocitiesBack(cars, braking);
```

OOP & DOD
In Memory





Object

Size =	23.1b
Useful Size =	10.1b
Used Size =	23.1b
Wasted =	13b

Key

- ID
- Colour
- Wheels
- BrakeForce
- Velocity
- IsBreaking

Entity

Size =	23.1b
Useful Size =	10.1b
Used Size =	10.1b
Wasted =	0

OOP & DOD Uses

- The Great War Engine MkI
- SandSim4000

What's Next?

- Wave Function Collapse
- Gamified Neurode Interface
- Job?

Townscaper – Oskar Stalberg

Age of Empires II
Ensemble Studios

Thank you!

Github

Youtube

Samuel Davidson